实验四 综合案例（2）

一、 实验目的

1、 掌握静态成员的定义

2、 掌握共享数据保护

二、 实验内容

1、PTA 作业《C++语言程序设计工程实践：面向对象基础》所有内容

2、使用 Visual Studio 环境来编辑、编译和运行下列程序。

(1) 在本实例中,将为 SavingsAccount 类增加一个静态数据成员 total,用来记录各个账户的总金额，并为其增加相应的静态成员函数 getTotal 用来对其进行访问。

(2) 在本实例中，将综合实例程序中 SavingsAccount 类的诸如 getBalance,accumulate 这些不需要改变对象状态的成员函数声明为常成员函数。

(3) 在本实例中将在综合实例的基础上对程序结构进行调整:将 SavingsAccount 类从主函数所在的源文件中分开,建立两个新的文件 account.h 和 account. cpp,分别存放 SavingsAccount 类的定义和实现。

//代码内容 1：类的静态成员与共享数据保护

```
//account.h
#ifndef __ACCOUNT_H__
#define __ACCOUNT_H__
class SavingsAccount { //储蓄账户类
private:
    int id;                 //账号
    double balance;         //余额
    double rate;        //存款的年利率
    int lastDate;       //上次变更余额的时期
    double accumulation;//余额按日累加之和
    static double total;    //所有账户的总金额
    //记录一笔帐，date 为日期，amount 为金额，desc 为说明
    void record(int date, double amount);
    //获得到指定日期为止的存款金额按日累积值
    double accumulate(int date) const {
        return accumulation + balance * (date - lastDate);
    }
public:
    //构造函数
```

```
        SavingsAccount(int date, int id, double rate);
        int getId() const { return id; }
        double getBalance() const { return balance; }
        double getRate() const { return rate; }
        static double getTotal() { return total; }
        void deposit(int date, double amount);        //存入现金
        void withdraw(int date, double amount);      //取出现金
        //结算利息，每年 1 月 1 日调用一次该函数
        void settle(int date);
        //显示账户信息
        void show() const;
    };
#endif //__ACCOUNT_H__


    //account.cpp
    #include "account.h"
    #include <cmath>
    #include <iostream>
    using namespace std;

    double SavingsAccount::total = 0;
    //SavingsAccount 类相关成员函数的实现
    SavingsAccount::SavingsAccount(int date, int id, double rate)
            : id(id), balance(0), rate(rate), lastDate(date), accumulation(0) {
        cout << date << "\t#" << id << " is created" << endl;
    }
    void SavingsAccount::record(int date, double amount) {
        accumulation = accumulate(date);
        lastDate = date;
        amount = floor(amount * 100 + 0.5) / 100;  //保留小数点后两位
        balance += amount;
        total += amount;
        cout << date << "\t#" << id << "\t" << amount << "\t" << balance <<
endl;
    }
    void SavingsAccount::deposit(int date, double amount) {
        record(date, amount);
    }
    void SavingsAccount::withdraw(int date, double amount) {
        if (amount > getBalance())
            cout << "Error: not enough money" << endl;
        else
            record(date, -amount);
```

```
}
void SavingsAccount::settle(int date) {
    double interest = accumulate(date) * rate / 365; //计算年息
    if (interest != 0)
        record(date, interest);
    accumulation = 0;
}
void SavingsAccount::show() const {
    cout << "#" << id << "\tBalance: " << balance;
}

//main.cpp
#include "account.h"
#include <iostream>
using namespace std;
int main() {
    //建立几个账户
    SavingsAccount sa0(1, 21325302, 0.015);
    SavingsAccount sa1(1, 58320212, 0.015);
    //几笔账目
    sa0.deposit(5, 5000);
    sa1.deposit(25, 10000);
    sa0.deposit(45, 5500);
    sa1.withdraw(60, 4000);
    //开户后第 90 天到了银行的计息日，结算所有账户的年息
    sa0.settle(90);
    sa1.settle(90);
    //输出各个账户信息
    sa0.show();  cout << endl;
    sa1.show();  cout << endl;
    cout << "Total: " << SavingsAccount::getTotal() << endl;
    return 0;
}
```

三、 实验要求

1、 完成 PTA 作业《C++语言程序设计工程实践：面向对象基础》的所有实验题。

2、 成功在 Visual Studio 2019 里面运行上述代码。